Tutorial on the Eclipse IDE

Norman Matloff, “Quick Tutorial on the Eclipse IDE Framework”, http://heather.cs.ucdavis.edu/~matloff/eclipse.html, accessed 19 Oct 2007

Working in C/C++:

- go to an existing project or make one; see "Projects" below
- edit your source file(s); see "Editing files" below
- if you don't already have the file **makefile**, create it (as an ordinary file, not a source file, i.e. **File | New | File**); it must have entries **all:** and **clean:**; e.g. for **z.c** in a Linux setting,

```
all: z

clean:
        /bin/rm z.o

z: z.o
        gcc -o z z.o

z.o: z.c
        gcc -g -c z.c
```

  of course, there are TABs in there; don't forget to save the file
- to compile, save your files if needed, right-click the project name, then select **Build Project**, which will run **make**
- if there were any compilation errors, they'll show up in several places:
  - in the **Console** view, which will show the result of the make
  - in the **Problems** view; if you click on any problem, the mouse pointer will be moved to the offending line in your editor view
  - in the editor view itself, where the compilation errors will be flagged with red disks just to the left of source code lines in your editor subviews; if you move the mouse pointer to one of the red disks, a yellow window will pop up to tell you what errors were found at that line
- if there were no compiler errors, run your code:
  - select **Run | Run...**; you will be asked for a run configuration
    - if you already have one for this program, click on it in the **Configurations:** view
    - if you don't have one, click on your language first, and then on **New**; choose a configuration name, and fill in your project and executable file names
  - set arguments, if any

    - hit **Run**
    - your program's output will appear in the **Console** view

  note that once you have your run configuration set up, you can bypass a step in future runs by selecting **Run | Run Last Launched**, or better yet, using the keyboard shortcut ctrl-F11
- if there were compiler errors, you can go to the error line either via using the mouse pointer in the editor, or by clicking on the error message in the **Problems** view
- to invoke the debugger, right-click on the project name, and select **Debug | Debug** ; highlight your executable name on the left, set any arguments, and hit **Debug**; you'll be asked whether you want to switch to the **Debug** perspective--say yes; see the Debugging: section below for operations
- to go back to edit/compile/run mode, go back to the **C/C++** perspective

Projects:

As with many IDEs, **Eclipse** organizes things into *projects*. A project is a directory within your workspace, which if you took the default when you installed **Eclipse** is the **workspace** directory within your **Eclipse** directory, e.g. **/usr/local/eclipse/workspace** or **$HOME/workspace**.

To make a new project:

- select **File | New | Project**
- select **C Project**
- hit Next; fill in the project name, then hit **Finish**
- a **New Project** dialog box will appear; just hit **Finish** unless you know the advanced stuff

To go to a project, either previously existing or one you just created: You can double-click the project name in the Navigator view or the C/C++ view, etc. If you don't see that view, select **Window | Show View** . You may have to right-click on the project name, and select **Open Project**.

If you are in the C/C++ perspective, there will also be a **C/C++ Projects** view, like **Navigator**'s.

Editing files:

Eclipse attempts to find an editor related to your file type (**.c**, **.py** etc.), possibly part of the plug-in for the language you're working in. It use syntax highlighting and perform other actions tailored to your file type.

To make a new file, say **z.c**:

- right-click on the project name, then select **New | Source File**

- make sure that the project name in **Source Folder** is correct; enter the file name, **z.c**; hit **Finish**
- the editor will now open a subview for editing **z.c**

To edit an existing file:

- go to **Navigator** (or **C/C++ Projects**, if doing C/C++)
- if the files don't appear, click on the arrow next to the project name
- double-click on the file you want

Noteworthy editing operations:

- copy-and-paste: note that text copied from a non-**Eclipse** views must still be pasted into **Eclipse** via **Edit | Paste**
- indenting/de-indenting: drag the mouse to highlight the lines of text to be moved right or left, then select **Edit | Shift Right/Left**; the default amount of space indented/de-indented can be configured by selecting **Window | Preferences** and then selecting the language
- importing from another directory: to copy some files from another directory, right-click on the project name and choose **Import**; choose **File system** and hit **Next**; fill in the **From directory** space, resulting in an icon appearing for that directory, with a check box; click the check box; uncheck the files you don't want to copy (or check **Deselect all** and then recheck the ones you want); select **Create selected folders only**; click **Finish**

Debugging:

Keep in mind that the behavior of the **Debug** perspective will vary from one language to another. But here are the features in common:

- you must be in the **Debug** perspective to debug (as explained earlier, **Run | Debug** will take you there if you are not already there)
- the **Debug** perspective has its own toolbar.
- to set a breakpoint, double-click in the gray column to the left of the source line
- to remove/modify a breakpoint (e.g. to make it conditional), right-click on its symbol
- breakpoints will be retained from session to session
- the current source code line (i.e. next line to be executed) will be highlighted; upon startup, that will be the first line in **main()** (or similar location, depending on the language)
- to resume execution (or start execution, if at the first line of **main()**), click on the resume icon at the left end of the debugger toolbar
- the current line of execution is highlighted in green
- the usual step-in/step-over operations are available

- there are views **Variables**, **Breakpoints** and so on, very useful
- while in the **Debug** perspective, to restart your program from the beginning, click the **Debug** icon in the **Eclipse** toolbar; or, if no change in arguments from before, F11 as a keyboard shortcut (**Debug Last Launched**)
- to return to your editing from the **Debug** perspective, click on the left arrow icon in the **Eclipse** toolbar (it goes to the previous perspective, like a Web browser going to the previous page)

Outlines:

Make good use of these! While you are editing, the **Outline** view will show you where your major code objects are, i.e. functions, classes, etc. Clicking on one will make the editor jump to that spot in the code.

The same is true for debugging. If you need to set a breakpoint at some function, for instance, click on its name in the outline, which will bring you right there, so you can set the breakpoint.

Resizing and deleting views:

Any IDE suffers from the problem that space on the screen tends to get cramped. **Eclipse** allows you to *maximize* (i.e. go full screen) any view, by clicking on the icon at the upper-right corner of the view.

To restore the original size, so you can see the rest of your **Eclipse** desktop, hit the interlocking-squares icon at the upper right.

You can delete a view or subview by clicking the big X at the upper-right corner of that view or subview.

Restoring views, etc.:

You may find the following reset mechanisms helpful:

- If a view is no longer on your screen, you can retrieve it by clicking **Window | Show Views** and then the view name.
- If within a perspective you wish to reset the views to their original configurations, select **Window | Reset Perspective**.
- To clear the **Console** and **Problems** views, etc., select **Project | Clean | Clean Selected Projects**
- To close everything but still stay in **Eclipse**, select **File | Close All**.

Using glut on the Eclipse IDE

To use glut and opengl then you need to get the compiler to link in the appropriate libraries.

Linking GL and GLUT libraries

- select **Project | Properties**.
- select **C/C++ Build | Settings**
- select **MingGW C Linker | libraries**
- Go to the Libraries (-l) panel. (Should be the the top one on the right)
    - click the **+** icon
        - type **glu32**
    - click the **+** icon
        - type **glut32**
    - click the **+** icon
        - type **open32**
- Click **Apply** and then **Ok**

For more information:

- There is a very extensive user manual (C/C++ focus, but still with good general information) at http://www.eclipse.org/cdt .
- There is a nice "Getting Started" introduction (Windows-based, but fairly general), at http://www.cs.umanitoba.ca/~eclipse .
- There is a PyDev tutorial at http://www-128.ibm.com/developerworks/opensource/library/os-ecant/ .
- GLUT for MinGW
- Nate Robins GLUT
- GLUT API - A valuable resource when programming with GLUT
- The Red Book (Code Samples) - recommended book if you want to learn how to program with OpenGL